EXHIBIT X

TO DECLARATION OF S. MERRILL WEISS IN SUPPORT OF PLAINTIFF ACACIA MEDIA TECHNOLOGIES CORPORATION'S MEMORANDUM OF POINTS AND AUTHORITIES IN OPPOSITION TO ROUND 3 DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. § 112 OF THE '992, '863, AND '702 PATENTS; AND SATELLITE DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY OF THE '992, '863, AND '720 PATENTS

Hovsepian v. Apple, Inc. Doc. 325 Att. 24

Dockets.Justia.com

Speech Coding Based Upon Vector Quantization

ANDRÉS BUZO, MEMBER, IEEE, AUGUSTINE H. GRAY, JR., SENIOR MEMBER, IEEE, ROBERT M. GRAY, FELLOW, IEEE, AND JOHN D. MARKEL, SENIOR MEMBER, IEEE

Abstract—With rare exception, all presently available narrow-band speech coding systems implement *scalar quantization* (independent quantization) of the transmission parameters (such as reflection coefficients or transformed reflection coefficients in LPC systems). This paper presents a new approach called *vector quantization.*

For very low data rates, realistic experiments have shown that vector quantization can achieve a given level of average distortion with 15 to 20 fewer bits/frame than that required for the optimized scalar quantizing approaches presently in use.

The vector quantizing approach is shown to be a mathematically and computationally tractable method which builds upon knowledge obtained in linear prediction analysis studies. This paper introduces the theory in a nonrigorous form, along with practical results to date and an extensive list of research topics for this new area of speech coding.

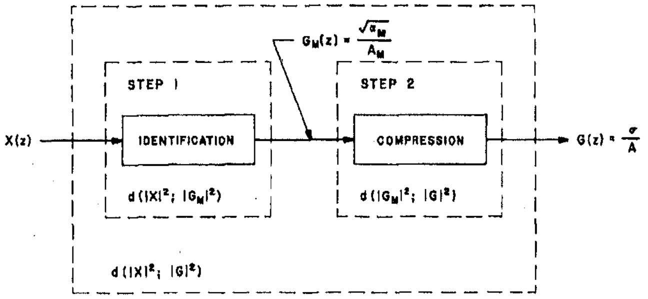

Fig. 1. Illustration showing process of LPC analysis as a two-step process.

I. Introduction

THE ubiquitous LPC technique of speech coding can be viewed as a two-step process as shown in Fig. 1. The first step is an identification process whereby an all-pole model $G_M(z)$ which best matches the input speech frame $X(z)$ (or possibly the preprocessed speech frame) is calculated. The best match is based on some predefined measure of optimality. The model $G_M(z)$ implicitly also includes a gain term so that the only remaining transmission parameter is the pitch estimate (which includes a voicing decision).

The second step is compression or quantization of the parameters from the identification step for efficient transmission or storage. A great deal has been written about the identification step (see [1], [2] and their bibliographies, for example), while substantially less has been written about the compression or quantization step [3]–[7].

Traditionally, the parameters from the identification step, such as reflection coefficients, have been individually quantized. We refer to such an approach as *scalar quantization* and it has also been called single symbol quantization. Orthogonal vector transformations have been applied with the idea of eliminating interparameter correlation. However, scalar quantization has then been applied to the transformed coefficients [8].

Other techniques, such as variable frame rate [9] or interframe coding, which can be "added on" to most speech compression systems, have been studied as a way to increase compression. To achieve highly efficient parameter compression, one must look beyond scalar quantization or the heuristics mentioned above to *rate distortion techniques* based on an appropriate fidelity criterion. Here, the fidelity criterion will include minimization of a distortion measure, not only in the identification step (as implemented in linear prediction analysis), but also in the compression or quantization step. *Vector quantization*, a design approach to this problem, has been developed during the past few years. The historical development of this area is covered in detail elsewhere [13], [14].

This paper presents a nonrigorous mathematical introduction to the new area of vector quantization in a deterministic manner along with experimental results. The presentation expands upon previously published results in linear prediction analysis. A more rigorous development of many of the theoretical concepts is presented elsewhere [10], [13], [18], [21]. In addition to the mathematical development, this paper presents the first experimental comparison between optimized scalar quantization and vector quantization. The results from both objective and subjective evaluation show dramatic bit savings for very low data rate conditions.

II. Itakura-Saito Distortion Measure

For a distortion measure to be of value in vector quantizing it must be analytically tractable, computable from sampled data, and, most important, subjectively meaningful. A distortion measure that results in the standard linear prediction

Manuscript received October 15, 1979; revised April 10, 1980. This research was supported in part by the Naval Research Laboratory under Contract NRLN00039-79-0256(S), the U.S. Department of Defense under Contract MDA904-79-C-0402, the U.S. Air Force Office of Scientific Research under Contracts F49620-78-C-0087 and F49620-79-C-0058, and by the Joint Services Electronics Program, Stanford University, Stanford, CA.

A. Buzo is with the University of Mexico, Mexico City, Mexico.
A. H. Gray, Jr. and J. D. Markel are with Signal Technology, Inc., Santa Barbara, CA 93101.
R. M. Gray is with the Information Systems Laboratory, Stanford University, Stanford, CA 94305.

0096-3518/80/1000-0562$00.75 © 1980 IEEE

Exhibit X Page 446

analysis equations under the assumption of no distortion due to compression is also desirable.

The *Itakura-Saito distortion measure*, introduced as an error matching function, appears to satisfy all of the above requirements [11]. This section reviews several preliminary mathematical results and introduces the Itakura-Saito distortion measure along with several properties relevant to specific vector quantizer implementations.

A. Preliminaries

Let $X(z)$ represent the z transform of the windowed (and possibly preemphasized) speech data, which are to be modeled by an all-pole filter of the form

$$G(z) \triangleq \sigma/A(z) \tag{1}$$

where

$$A(z) \triangleq \sum_{k=0}^{M} a_k z^{-k}, \quad \text{with } a_0 = 1. \tag{2}$$

In linear predictive analysis the polynomial $A(z)$ is used to minimize a residual energy. In particular, using $|X|^2$ and $|A|^2$ to denote energy density spectra

$$|X|^2 \triangleq |X(e^{j\theta})|^2 \quad \text{and} \quad |A|^2 \triangleq |A(e^{j\theta})|^2, \tag{3}$$

then the residual energy resulting from passing $X(z)$ through the inverse filter $A(z)$ is given by

$$\alpha \triangleq \int_{-\pi}^{\pi} |X|^2 \, |A|^2 \, \frac{d\theta}{2\pi}. \tag{4}$$

We denote the polynomial which minimizes the residual energy as $A_M(z)$ and the minimum value of the residual as α_M so that

$$\alpha \geqslant \alpha_M.$$

The model chosen in linear prediction analysis based upon the identification step in Fig. 1 is then

$$G_M(z) = \sqrt{\alpha_M}/A_M(z). \tag{5}$$

It is well known [1] that $A_M(z)$ is a minimum phase polynomial, that is, has its roots inside the unit circle so that $G_M(z)$ is stable. In addition, the minimum residual energies form a decreasing sequence as M increases, approaching a lower limit defined here as α_∞, which is given by

$$\alpha_\infty = \lim_{M\to\infty} \alpha_M = \exp\left[\int_{-\pi}^{\pi} \ln |X|^2 \frac{d\theta}{2\pi}\right]. \tag{6}$$

This lower limit α_∞ is sometimes called the one-step prediction error or gain of $|X|^2$ [1].

The actual solution process for finding $A_M(z)$ is also well known (see [1] and the references contained therein) and is not considered here.

Using arrows to denote z transform relationships, we can define the autocorrelation sequences

$$X(z)\,X(1/z) \longleftrightarrow r_x(n) = \sum_k x(k)\,x(k+n) \tag{7a}$$

$$A(z)\,A(1/z) \longleftrightarrow r_a(n) = \sum_k a_k a_{k+n} \tag{7b}$$

$$G_M(z)\,G_M(1/z) \longleftrightarrow r_M(n). \tag{7c}$$

Limits have not been placed on the summations to indicate that they are summations over all values of k. As will be pointed out, however, these will always be finite summations.

The integral which defines the residual energy in (4) can be precisely expressed for numerical evaluation as

$$\alpha = \sum_n r_x(n)\, r_a(n). \tag{8}$$

It is well known [1] that the model $G_M(z)$ matches the signal $X(z)$ in terms of the $2M+1$ term autocorrelation sequence

$$r_M(n) = r_x(n) \quad \text{for } n = 0, \pm 1, \cdots, \pm M, \tag{9}$$

and therefore $r_M(n)$ can be used to replace $r_x(n)$ in the summation of (8).

In describing the spectral matching effects of linear prediction, Itakura and Saito introduced an "error matching function" [11] which is also referred to as the Itakura-Saito distortion measure [10]. This measure will be denoted by $d(|X|^2; |G|^2)$, where

$$d(|X|^2; |G|^2) \triangleq \int_{-\pi}^{\pi} [|X/G|^2 - \ln(|X/G|^2) - 1] \frac{d\theta}{2\pi}. \tag{10}$$

Many of its properties can be found elsewhere [10]–[12], [18]. One of the properties that interests us here is that it is a nonnegative function of the spectra, whose minimum value for a given $|X|^2$ and $G(z)$ given by (1) occurs when $G(z) = G_M(z)$. Thus,

$$d(|X|^2; |G|^2) \geqslant d(|X|^2; |G_M|^2) = \ln(\alpha_M/\alpha_\infty). \tag{11}$$

When $|X/G|$ is near unity, (10) takes on the approximate form

$$d(|X|^2; |G|^2) \cong \frac{1}{2}\int_{-\pi}^{\pi} [\ln(|X|^2) - \ln(|G|^2)]^2 \frac{d\theta}{2\pi}, \tag{12}$$

so that for small distortion the Itakura-Saito distortion measure is approximately one-half the mean-square log spectral deviation.

B. Useful Properties

A set of properties of the Itakura-Saito distortion measure to be used in the next section is now presented. A sketch of the derivation of these results is given in the Appendix. The properties are developed in detail in [10] and [12].

First, for purposes of calculation and interpretation, the Itakura-Saito distortion measure can be expressed in the form

$$d[|X|^2; |G|^2] = \alpha/\sigma^2 + \ln(\sigma^2) - \ln(\alpha_\infty) - 1 \tag{13}$$

where σ, α, and α_∞ are defined by (1), (4), and (6), respectively. Surprisingly, it can be shown to satisfy a form of "triangle equality"

$$d[|X|^2; |G|^2] = d[|X|^2; |G_M|^2] + d[|G_M|^2; |G|^2], \tag{14}$$

Exhibit X Page 447

as shown in the Appendix. Referring to Fig. 1 we see that *the total distortion in the analysis can be viewed as precisely the sum of the distortion due to the identification step and the distortion due to the compression or quantization step.* In general, one can only hope for a triangle inequality whereby the total distortion is bounded by the sum of the individual distortions. Also, minimizing $d[|X|^2; |G|^2]$ is equivalent to minimizing $d[|G_M|^2; |G|^2]$, for $d[|X|^2; |G_M|^2]]$ is a fixed property of $|X|^2$, as indicated in (11), if M is held constant.

A second useful gain cascading property is given by

$$d[|X|^2; \sigma^2/|A|^2] = d[|X|^2; \alpha/|A|^2] + d[\alpha; \sigma^2], \qquad (15)$$

which divides the distortion into two parts. The first part is independent of the gain choice σ, while the second part is apparently independent of the polynomial $A(z)$, although in fact it does depend upon $A(z)$ through the residual energy α. The first term has also been called the gain optimized Itakura-Saito distortion measure or simply the "Itakura distortion measure" [10], [12]. We will show later that this form leads to implementation of a suboptimal but storage-efficient and a practical narrow-band speech compression system.

III. Vector Quantization

The standard autocorrelation method LPC system first obtains an optimal model $G_M(z)$ for a speech frame $X(z)$, and then quantizes its parameters, leading to a quantized version $G(z)$. This system implicitly minimizes the Itakura-Saito distortion measure [1, p. 135], [10], [11] at the first step (identification of the optimal model), but does not use this distortion measure for the second step (compression or quantization of the parameters). It is therefore unknown whether any particular overall distortion from the speech spectrum $|X|^2$ to the quantized model spectrum $|G|^2$ has in fact been minimized.

An alternate approach is to define a reasonable distortion measure and attempt to choose $G(z)$ from a finite collection of vectors to minimize the overall distortion. Thus, the term vector quantization refers to the process of choosing a vector of parameters, e.g., $\{\sigma, r_a(0), r_a(1), \cdots, r_a(M)\}$, from a set which minimizes a distortion measure. As described in the previous section, the Itakura-Saito distortion measure is analytically tractable, perceptually meaningful, and readily computable (except for the term α_∞ in (13) which will be shown to be unnecessary).

One very important and useful property of the distortion measure is the triangle equality of (14). In particular, this property shows that one can minimize the overall distortion $d(|X|^2; |G|^2)$ directly in one step, or one can first obtain the ideal model $G_M(z)$ and then minimize $d(|G_M|^2; |G|^2)$, which results in a two-step process as indicated in Fig. 1 [10].

Aside from computational and storage issues, the vector quantizer being described here results in an elegant structure (shown later), once the nontrivial problem of how to determine a collection of reference or reproduction vectors or *codebook* has been resolved. The codebook is a finite set of model filters from which $G(z)$ must be found. Each code word has an index number and a stored set of parameters representing a possible $G(z)$. For example, code word number 01101001 (binary) might represent one frame of the sound /a/ as uttered by a specific test speaker, by way of its filter coefficients, reflection coefficients or autocorrelation coefficients. Then for each frame of speech, $G(z)$ is chosen from the codebook to minimize $d(|X|^2; |G|^2)$, or, equivalently, to minimize $d(|G_M|^2; |G|^2)$. This process finds the nearest neighbor to $X(z)$ in the codebook. The index to the code word is then transmitted and used at the receiver to retrieve the appropriate synthesis filter parameters.

The problems of generating an optimal codebook (in the Itakura-Saito distortion sense) and evaluating the distortion are now analyzed. Then the computational and storage issues which lead to a practical suboptimal vector quantizing system are considered.

A. Codebook Generation

The generation of a codebook to minimize distortion over a large number of test frames of speech requires an iterative process. As with many minimization problems, the procedure to be described will converge to a local minimum, but not necessarily an absolute or global minimum. The basic ideas of the procedure to be described date to Lloyd [15]. Linde *et al.* [13] and Gray *et al.* [21] contain a more detailed and general discussion.

Let $X_k(z)$ represent the z transform of the kth frame of speech, where $k = 1, 2, \cdots, K$, and K is large. These frames represent a test or learning sequence for codebook generation. We wish to model these speech frames with a finite set of models, $B = 2^b$ in number, where b is the number of bits in the codebook. Assume at this point that an initial (nonoptimum) choice has been made for a set of B code words. As a first step, each speech frame, represented by $X_k(z)$, is taken individually and assigned to a "cell" represented by a single code word. This is accomplished by finding the particular $G(z)$ out of the collection of B possible models which minimizes $d[|X_k|^2; |G|^2]$. That is, the objective is to find the particular $G(z)$ in the codebook which is the nearest neighbor to $X_k(z)$. The total overall distortion is then the sum of all the individual distortions in the database, with one from each speech frame.

The second step attempts to improve the codebook by choosing a better model for each cell. For simplicity, consider a single cell, with the frames of speech within that cell renumbered as $X_1(z), X_2(z), \cdots, X_L(z)$, so that the total distortion for that cell is given by

$$D \triangleq \sum_{k=1}^{L} d[|X_k|^2; |G|^2]. \qquad (16)$$

Then we must find a single model $G(z)$ from an infinite number of possibilities to minimize the distortion in that particular cell by identifying what might be called the *centroid* of the cell. The solution to this problem is equivalent to a standard linear prediction analysis problem of minimizing $d[|\overline{X}|^2; |G|^2]$, where $|\overline{X}|^2$ is the arithmetic mean of the individual cell spectra

$$|\overline{X}|^2 \triangleq \frac{1}{L} \sum_{k=1}^{L} |X_k|^2. \qquad (17)$$

Exhibit X Page 448

Once the centroids of each cell have been found and used as the new code words (model filters), the distortion for each individual cell is minimized by the definition of a centroid. Thus the overall distortion must decrease, or at worst remain the same. One then returns to the first step to redefine the cells by assigning each speech frame to its nearest neighbor model filter. By definition of a nearest neighbor, the distortion for each individual speech frame and thus the total distortion must decrease, or at worst remain the same. Iteration of this procedure results in an overall distortion that is monotonically nonincreasing, for at each step it either decreases or remains the same. As the distortion is bounded below, it must approach a lower limit—a locally optimum value [21]. The speed of convergence and the actual value of the local minimum of the overall distortion will depend upon the initial choices for the code words at the start of the iteration.

B. Nearest Neighbor Distortion Calculation

To assign a frame of speech to a specific code word, we must find the specific $G(z)$ out of the collection of all possible models which minimizes $d[|X|^2; |G|^2]$. From (13), we note that since α_∞ depends only on the speech frame, an equivalent statement is to find the $G(z) = \sigma/A(z)$ which will minimize

$$d[|X|^2; |G|^2] + 1 + \ln(\alpha_\infty) = \alpha/\sigma^2 + \ln(\sigma^2).$$

For any single frame of speech, the residual energy α must be computed. The computation is most efficiently accomplished in the form (see the Appendix)

$$\alpha = r_a(0)\, r_x(0) + 2 \sum_{n=1}^{M} r_a(n)\, r_x(n) \qquad (18)$$

where $r_a(n)$ and $r_x(n)$ are the autocorrelations of (7). The computational expression for $r_a(n)$ is given by

$$r_a(n) = \sum_{k=0}^{M-n} a_k a_{k+n} \qquad \text{for } n = 0, 1, \cdots, M. \qquad (19)$$

For computational efficiency, the transmitter codebook should probably contain the normalized sequence $\{r_a(n)/\sigma^2, n = 0, 1, \cdots, M\}$, as well as the value of $\ln(\sigma^2)$, so that $\alpha/\sigma^2 + \ln(\sigma^2)$ can be evaluated with exactly $M+1$ multiplies and $M+2$ adds, after $r_x(n)$ is computed. The transmitter codebook then contains $M+2$ parameters for each code word, rather than $M+1$ parameters, which is probably a desirable tradeoff to eliminate the division by σ^2 and/or the log calculation $\ln(\sigma^2)$.

In evaluating (18), $r_x(n)$ is required for $n = 0, 1, \cdots, M$. This is the standard short-term autocorrelation sequence for the sequence $x(n)$ used in the autocorrelation method of linear prediction. For a data sequence which is truncated to $n = 0, 1, \cdots, N-1$ samples, $r_x(n)$ is given by

$$r_x(n) = \sum_{k=0}^{N-1-n} x(k)\, x(k+n) \qquad \text{for } n = 0, 1, \cdots, M < N. \qquad (20)$$

From the autocorrelation matching property, the term $r_x(n)$ can be replaced by its equivalent model autocorrelation term $r_M(n)$. This is convenient for simulation studies where test data may be stored only in the format of the model $G_M(z)$.

In summary, to find the nearest neighbor one must evaluate $\alpha/\sigma^2 + \ln(\sigma^2)$ for each entry in the codebook, and choose the code word which minimizes the result. Therefore, this procedure requires an exhaustive search for every speech frame.

C. Centroid Calculation

If the frames of speech $X_1(z), X_2(z), \cdots, X_L(z)$ are all contained within a given cell, the total distortion for that cell is given by (16), which can also be written in terms of the average spectrum of (17) as

$$D = L\, d[|\overline{X}|^2; |G|^2] + u \qquad (21)$$

where u is a constant that is independent of $G(z)$, the model for the cell. Thus to find the cell centroid (the $G(z)$ that minimizes D) we have a standard linear prediction problem of modeling the average spectrum.

As a result, we can average the autocorrelation sequences for each of the speech frames to find an average autocorrelation sequence, and then solve the autocorrelation equations to give the parameters of $G(z) = \sigma/A(z)$. The constant u is not needed for these calculations. However, it represents a cost or distortion that will arise, regardless of model filter order, when dissimilar frames of speech are assigned to the same cell. See the Appendix for its exact form.

D. Speech Coder Implementation

Figs. 2 and 3 show a new speech coder structure based on the material of Section III. The system implements a *full search optimal vector quantizer*, operating directly on the speech waveform.

The analysis process consists of calculating an $M+1$ length autocorrelation sequence and then running a full search comparison of the codebook to obtain the index $i_{\min}$ for which $\alpha/\sigma^2 + \ln(\sigma^2)$ results in minimum distortion. Using any of the standard methods of computing a pitch and voicing term with corresponding transmission index i_p, the analysis is completed.

At the synthesizer, $i_{\min}$ is inserted into the receiver codebook to obtain the synthesis parameters in a form ready for use by the synthesis structure. For example, although $M+2$ terms $\{\ln(\sigma^2), r_a(0)/\sigma^2, \cdots, r_a(M)/\sigma^2\}$ define the analyzer code word, the corresponding synthesis code word would most likely contain the $M+1$ terms $\{\sigma, k_1, \cdots, k_M\}$ for use with a lattice form synthesizer.

E. Discussion

An extremely interesting aspect of this development is that *no* simultaneous equations are solved for reflection coefficients or filter coefficients, even though our process is now optimal (in the Itakura-Saito sense) not only for the identification step (as in standard LPC analysis), but also for the compression or quantization step. In effect, the speech processing system based on full search optimal vector quantization becomes a one-step combined identification and compression step as shown in Fig. 4, with full optimality throughout.

Once the speech codebook has been obtained, several difficulties remain. The first problem relates to storage of a sub-

Exhibit X Page 449

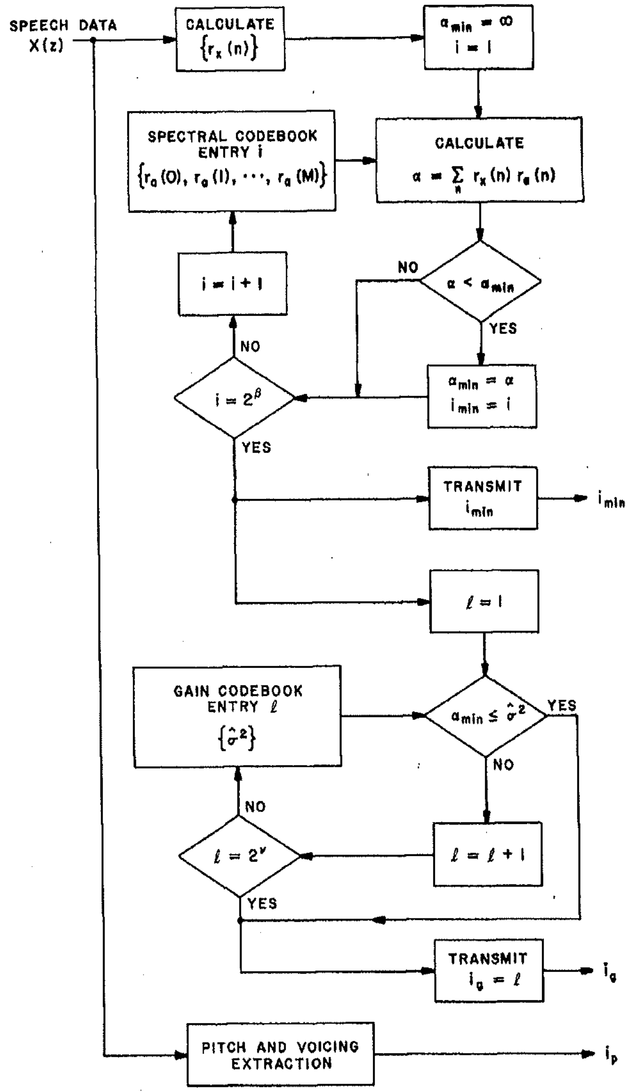
SPEECH DATA
X(z)
CALCULATE
{r_x(n)}
α_min = ∞
i = 1
SPECTRAL CODEBOOK ENTRY i
{r_a(0), r_a(1), ···, r_a(M)}
CALCULATE
α = Σ r_x(n) r_a(n)
i = i + 1
NO
α < α_min
YES
NO
i = 2^β
α_min = α
i_min = i
YES
TRANSMIT
i_min
i_min
ℓ = 1
GAIN CODEBOOK ENTRY ℓ
{σ̂²}
α_min ≤ σ̂²
YES
NO
NO
ℓ = 2^ν
ℓ = ℓ + 1
YES
TRANSMIT
i_g = ℓ
i_g
PITCH AND VOICING EXTRACTION
i_p

Fig. 2. Analysis structure for full search optimal vector quantized speech processor.

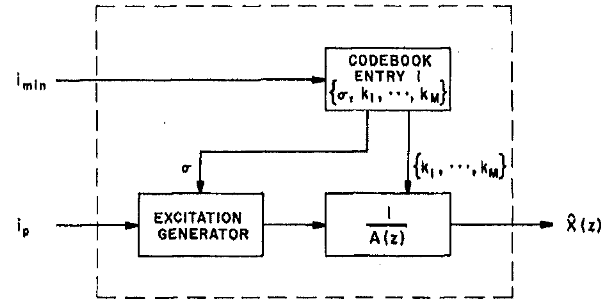
i_min
CODEBOOK ENTRY i
{σ, k_1, ···, k_M}
σ
{k_1, ···, k_M}
i_p
EXCITATION GENERATOR
1/A(z)
X̂(z)

Fig. 3. Synthesis structure for full search optimal vector quantized speech processor.

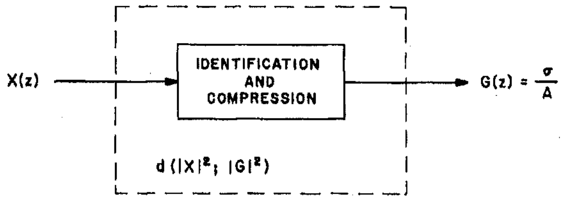
X(z)
IDENTIFICATION AND COMPRESSION
G(z) = σ/A
d(|X|²; |G|²)

Fig. 4. A one-step identification and compression system.

stantial amount of information. Since both gain and spectral information are coded together, the codebook is larger than if they were coded independently. For example, if 15 bits are used to represent the combination of spectral and gain information for model filters of order $M = 10$, then a total of at least $2^{15} = 32\,768$ code words times 12 coefficients per code word equals 393 216 storage locations required at the transmitter and receiver. In general, $(M + 2)\, 2^b$ storage locations are required with this procedure as it is presently structured. Although this amount of storage is feasible with a general-purpose computer having virtual memory, such a requirement is unrealistic for 16 bit minicomputers and special purpose real-time hardware.

A second problem relates to the amount of computation required in the determination of a code word index. Each speech frame must be compared with $B = 2^b$ possible models to choose a nearest neighbor. For example, a 15 bit codebook would require a total of 2^{15} evaluations of $\alpha/\sigma^2 + \ln(\sigma^2)$, with each evaluation requiring at least $M + 1$ multiplies. The number of these calculations grows exponentially with the number of bits.

IV. Suboptimal Vector Quantization

The rather sobering facts just described in the previous section would render the vector quantizing approach of only academic interest for codebooks greater than 10 bits were it not for two major modifications that we have developed which result in a suboptimal vector quantization system. The first modification is termed gain separation, and the second modification is termed binary tree searched coding. The latter is, to the best of our knowledge, the first suggested approach for vector searching with binary search savings. This section considers these topics. The previously described optimal system using full search and combined gain and model code words is developed in more mathematical and experimental detail in [18].

A. Gain Separation

Equation (15) illustrates a separation of the distortion into two parts. The first depends only upon the polynomial $A(z)$—not upon the gain σ—and the second depends upon the gain term (and indirectly on the polynomial through α). Rather than minimize the overall distortion, one can separately minimize the two parts of (15) by first finding $A(z)$ and then obtaining σ. The overall codebook is suboptimal since it is restricted to have a particular form. However, this approach can result in a decreased codebook size at the expense of increased overall distortion. For example, if β bits are assigned to describe $A(z)$ and ν bits are assigned to describe σ, the number of words in the codebooks for the separated parameters are reduced from $2^{\beta+\nu}$ to $2^\beta + 2^\nu$ for a full codebook containing $\beta + \nu$ bits. As will be shown, this separation leads to an approximation (further suboptimization) in the calculation of centroids needed to generate the codebook for the polynomial parameters.

Exhibit X Page 450

1) *Nearest Neighbor Calculations*

For any given speech frame $X(z)$ we must first find the nearest neighbor in the $A(z)$ codebook which minimizes $d[|X|^2; \alpha/|A|^2]$ of (15). Substituting $\sigma^2 = \alpha$ in (1) and (13) gives an equivalent expression

$$d[|X|^2; \alpha/|A|^2] = \ln(\alpha) - \ln(\alpha_\infty). \tag{22}$$

This expression can therefore be minimized by minimizing α, the residual energy. As in Section III-A, α can be calculated by using (18). The calculation of α requires $M + 1$ multiply/add's for each entry in the codebook to determine which one gives the minimum value for a given speech frame.

Once the minimum α has been found, one then looks to the gain codebook to find which gain value will minimize the remaining part of (15), $d(\alpha; \sigma^2)$. Substituting α and σ^2 into the distortion measure (10) directly gives

$$d(\alpha; \sigma^2) = (\alpha/\sigma^2) - \ln(\alpha/\sigma^2) - 1, \tag{23}$$

which is minimized by choosing a value of σ^2 from the gain portion of the codebook. The same result can be arrived at more expeditiously by noting from (13) that for any value of gain σ^2 the overall distortion is minimized by minimizing the residual energy α, and for any value of α, the overall distortion is minimized by choosing the gain to minimize $\alpha/\sigma^2 + \ln(\sigma^2)$, or equivalently by minimizing (23). For a given product codebook, this approach gives the optimal choice of a code word. The codebook itself is suboptimal, not the assignment of the code word to a speech frame.

Since minimization of (23) is a one-dimensional problem, the codebook gains can be ordered and compared with a set of threshold values which will be stored at the transmitter, with the codebook gains stored at the receiver.

To illustrate this point, subscripts are used on the codebook gains (assumed ordered), so that the codebook consists of σ_l^2 for $l = 1, 2, \cdots, \Upsilon$, with

$$\sigma_1^2 < \sigma_2^2 < \cdots < \sigma_\Upsilon^2.$$

We shall find a set of thresholds, or cell boundaries $\hat{\sigma}_l^2$, $l = 1, 2, \cdots, \Upsilon - 1$, such that

$$\sigma_1^2 < \hat{\sigma}_1^2 < \sigma_2^2 < \hat{\sigma}_2^2 < \cdots < \hat{\sigma}_{\Upsilon-1}^2 < \sigma_\Upsilon^2.$$

At the transmitter the set of $\Upsilon - 1$ thresholds is stored, and α is compared to the thresholds to find which cell it lies in to generate an index for transmission. For example, if $\alpha \leqslant \hat{\sigma}_1^2$, the index $l = 1$ is transmitted, or if $\hat{\sigma}_4^2 < \alpha \leqslant \hat{\sigma}_5^2$, then the index $l = 5$ is transmitted. At the receiver, the index l is used to recover the stored gain term σ_l^2.

Finding the threshold values from the codebook gains is a matter of solving the equations

$$d(\hat{\sigma}_l^2; \sigma_l^2) = d(\hat{\sigma}_l^2; \sigma_{l+1}^2) \tag{24}$$

for $\hat{\sigma}_l^2$, the point "equidistant" between adjacent codebook gains. This must be done for $l = 1, 2, \cdots, \Upsilon - 1$. To solve this equation, we can use (23) to rewrite (24) in the form

$$(\hat{\sigma}_l^2/\sigma_l^2) - \ln(\hat{\sigma}_l^2/\sigma_l^2) - 1 = (\hat{\sigma}_l^2/\sigma_{l+1}^2) - \ln(\hat{\sigma}_l^2/\sigma_{l+1}^2) - 1, \tag{25}$$

which can then be solved to give

$$\hat{\sigma}_l^2 = \frac{\ln(\sigma_{l+1}^2/\sigma_l^2)}{(1/\sigma_l^2) - (1/\sigma_{l+1}^2)}. \tag{26}$$

When σ_{l+1}^2 is near σ_l^2, as it would be if there is a fine-level quantization, then (26) is not numerically efficient because of the subtraction of close numbers. In that case it is better to use a Taylor series expansion to write

$$\hat{\sigma}_l^2 = \frac{1}{2}(\sigma_l^2 + \sigma_{l+1}^2)\left[1 - \frac{2\delta^2}{3 \cdot 1} - \frac{2\delta^4}{5 \cdot 3} - \frac{2\delta^6}{7 \cdot 5} - \cdots\right] \tag{27a}$$

where

$$\delta \triangleq (\sigma_{l+1}^2 - \sigma_l^2)/(\sigma_{l+1}^2 + \sigma_l^2). \tag{27b}$$

The derivation of the Taylor series is omitted for brevity. The use of (27) rather than (26) is strongly suggested for numerical accuracy.

In summary, the threshold values can be obtained from the code word gains using (27) or (26). Finding the code index for gain requires using the residual energy α, which was obtained during the coding of the polynomial $A(z)$ in a direct comparison with the threshold values.

2) *Centroid Calculation*

In the gain separated case we are dealing with two centroids to be calculated. First, for the polynomial parameters, we would like to minimize the total cell distortion of (16). Using (15) and (22) we are attempting to minimize the sum of terms

$$D_1 = \sum_{k=1}^{L} d[|X_k|^2; \alpha^k/|A|^2] = \sum_{k=1}^{L} [\ln(\alpha^k) - \ln(\alpha_\infty^k)] \tag{28}$$

where each α^k is the "optimal" gain choice for the individual speech frames (*not* the kth power of α),

$$\alpha^k = \int_{-\pi}^{\pi} |X_k|^2 \, |A|^2 \frac{d\theta}{2\pi},$$

and α_∞^k is the one-step prediction error or gain of $|X_k|^2$ as in (6). Thus, the centroid problem is to choose a polynomial $A(z)$ to minimize

$$\sum_{k=1}^{L} \ln(\alpha^k) = \sum_{k=1}^{L} \ln\left[\int_{-\pi}^{\pi} |X_k|^2 \, |A|^2 \frac{d\theta}{2\pi}\right].$$

Clearly, finding the polynomial $A(z)$ to minimize (28) is not a trivial task. Instead of trying to solve the specific problem, we shall look to an approximate (and bounding) solution.

Each individual $X_k(z)$ has an "optimal" model whose gain or one-step prediction error term is given by α_M^k. Equation (28) is first rewritten in the form

$$D_1 = \sum_{k=1}^{L} \ln(\alpha^k/\alpha_M^k) + \sum_{k=1}^{L} \ln(\alpha_M^k/\alpha_\infty^k). \tag{29}$$

The second summation is *independent of the parameters of the polynomial* $A(z)$, and is simply a function of the individual speech frames. The first summation of (29) is the product of

Exhibit X Page 451

L and the logarithm of the geometric mean of the ratios α^k/α_M^k for $k = 1, 2, \cdots, L$. An approximation to the geometric mean, and an upper bound as well, is given by the arithmetic mean so that D_1 is both approximated by and bounded above by D_2, where

$$D_2 = L \ln \left[\frac{1}{L} \sum_{k=1}^{L} (\alpha^k/\alpha_M^k) \right] + \sum_{k=1}^{L} \ln (\alpha_M^k/\alpha_\infty^k). \tag{30}$$

To minimize D_2 exactly, and thus D_1 approximately, we need to minimize the arithmetic mean of the α^k/α_M^k ratio defined by

$$\frac{1}{L} \sum_{k=1}^{L} (\alpha^k/\alpha_M^k) = \int_{-\pi}^{\pi} |\overline{\overline{X}}|^2 \, |A|^2 \frac{d\theta}{2\pi} \tag{31}$$

where $|\overline{\overline{X}}|^2$ is a normalized average spectrum given by

$$|\overline{\overline{X}}|^2 \triangleq \frac{1}{L} \sum_{k=1}^{M} |X_k|^2/\alpha_M^k. \tag{32}$$

Thus, centroid determination in the gain-separated case is similar to that in the optimal case, with the basic differences being that it is actually an approximation in the present case, and that the individual spectra (or autocorrelation sequences) are normalized before averaging.

In summary, one can average the autocorrelation sequences for all the speech frames within a cell and solve the autocorrelation equations. In the fully optimal case one does not normalize the autocorrelation sequence, but in the gain-separated case, one must normalize the autocorrelation sequences by the optimal gain coefficients, the α_M^k terms obtained from the residual energy resulting from passing the speech frame through its optimal inverse filter. In this latter case, frames of silence will count as heavily in finding a centroid as frames of high-energy voiced speech since the autocorrelation sequences are normalized. This suggests a careful screening of any reference frames used in generating the codebook.

Finding the centroid for the gain codebook is simpler, once the α^k for each frame has been found, for then one is interested in choosing a single gain term σ so as to minimize

$$D_3 = \sum_{k=1}^{L} d[\alpha^k; \sigma^2] = \sum_{k=1}^{L} [(\alpha^k/\sigma^2) - \ln (\alpha^k/\sigma^2) - 1]. \tag{33}$$

This expression can be minimized by taking σ^2 as the arithmetic mean of the individual residual energies

$$\sigma^2 = \frac{1}{L} \sum_{k=1}^{L} \alpha^k.$$

B. Binary Tree Searched Coding

To decrease the number of calculations necessary for finding a nearest neighbor code word, one can institute a binary search at the expense of doubling the storage requirements at the transmitter and increasing distortion. The systems described so far, with or without gain separation, require a large number of calculations for each speech frame to obtain a nearest neighbor growing exponentially with the number of bits. In the alternative described here and used in the experimental results, the number of calculations grows only linearly with the number of bits, but with twice the amount of storage at the transmitter and some increase in the distortion.

The codebook at the transmitter is split into levels. The first level contains only two code words and is used to split the space of speech frames into two, by one pair of residual energy evaluations and a comparison. Then each of these large cells is split into two, with a total of four code words at the second level. As we are seeking practical suboptimal systems, we will assume that only the parameters of the polynomial $A(z)$ are being treated here. Thus, if the polynomial parameters are quantized to a total of β bits, there must be a total of

$$2 + 2^2 + \cdots + 2^\beta = 2^{\beta+1} - 2$$

stored code words at the transmitter, though there are still only 2^β at the receiver. In effect, the transmitter storage is doubled for the part of the codebook holding parameters for the polynomial $A(z)$.

The number of calculations is decreased substantially in this case. Where 2^β residual energy evaluations were needed before to find a nearest neighbor, that number is now reduced to 2β.

Using the same size training sequence, calculation of the suboptimal approach is similar to that of the optimal approach except for a substantial reduction in computer time. This is because the present initialization procedures for the codebooks involve a binary splitting (see Section V), which uses all members of the training sequence at each step. For the suboptimal binary tree search procedure, one first sets up a 1 bit codebook, effectively splitting the training sequence in two. While the two halves do not have an equal number of members, they each have less than the original number, and each half is then split in two, setting up a 2 bit codebook. This procedure continues until a β bit codebook is finally established, requiring $2^{\beta+1} - 1$ code words at the transmitter, yet still only 2^β code words for the filter parameters at the receiver.

In this form of binary system with the gain separated from the model, the codebook for gain is generated as described earlier, with no change, from the resulting residual energies from the training sequence speech frames.

C. Initialization

As the codebook is at best locally optimal (or suboptimal for the gain-separated and/or binary tree searched case), the choice of initial code words can be significant in terms of the final outcome. We shall address ourselves here to the binary suboptimal coding as described in the preceding section, but many of the problems are common to the more fully optimal situation.

In the binary situation, the problem is to choose two initial code words for splitting a specific cell since this is the way the space is split.

One approach is to find the centroid of the cell, and then perturb it in some manner to give two different points. This is an ad hoc approach not guaranteed to be better than others, but it has given us reasonably satisfactory results. In particular, we find the reflection coefficients associated with the centroid, and then generate two code words by multiplying

Exhibit X Page 452

the reflection coefficients by the arbitrary factors of 1.01 and 0.99, respectively. From that point the iteration further separates the code words and "splits" the cell.

An elementary alternative to this procedure is to first find the frame in the cell farthest from the centroid (in the sense of the Itakura-Saito distortion measure) to use as an initial code word. Then one finds the frame furthest from that code word and uses that as the second initial code word. In this case the initial pair of code words are far apart rather than close together. Only experimentation will determine which approaches are more efficient in the sense of number of needed iterations and size of the local minimum that results.

It remains to be seen as to what the "best" initialization is, and how to handle a few of the problems. One problem that arises is the "empty cell." For example, if one has generated a 9 bit codebook, where only one frame of speech from the training sequence is assigned to a cell (with zero quantization distortion), there is no way to split that particular cell. This condition leads to an inefficient use of the codebook, as well as to possible error messages if one is not careful with the programming. This problem is related to the training set size.

One should have a reasonable number of training samples (speech frames) for each cell, such as at least 10 or 20, so that samples outside the training set are reasonably represented. If a reference set not exceeding $B = 2^b$ samples is chosen, zero distortion can theoretically be obtained, but only for the reference set. For a 12 bit codebook, a training sequence of 40 000-80 000 speech frames should be used. With reasonable computation speed a large number of reference frames can be obtained (in a recent study 7×10^6 disk based speech frames were obtained from conversational speech [17]). However, *substantial* amounts of computing are required to determine the codebook because of the interactive process described above (several representative times are presented in the section on experimental results).

D. *Suboptimal Speech Coder Implementation*

A speech processing system which implements suboptimal vector quantizing is shown in Fig. 5 (analyzer) and Fig. 6 (synthesizer). The process is quite similar to full search optimal vector quantizer implementation except that two separate codebooks are used in the analyzer and synthesizer, and the spectral parameter codebook is searched in a binary as opposed to a full search manner.

No detailed formulas have been given in this section on the suboptimal gain separated system since they have already been presented and are indicated again in Fig. 5. A few points will be emphasized, however. First, the only normalization used in the suboptimal system is in the evaluation of centroids needed to set up the codebook. No normalization is needed to use the codebook. Second, while it was computationally efficient to store the parameters $r_a(k)/\sigma^2$, for $k = 0, 1, \cdots, M$ for the fully optimal case where an evaluation of α/σ^2 was required, the suboptimal case requires only α.

For a 12 bit spectral codebook and $M = 10$ spectral coefficients, there are $2\beta = 24$ comparisons, each requiring $M + 1 = 11$ multiply/add's for a total of 264 multiply/add's. This number is essentially equivalent to the approximately $2.5M^2 = 250$ operations required to perform a single frame linear prediction analysis solution. Thus, we now have a structure that can be considered for real-time implementation with at least up to 12 bits/frame. The next section presents a set of preliminary results which compare distortion as a function of bits/frame for the new suboptimal vector quantizer with the standard scalar quantizer approach.

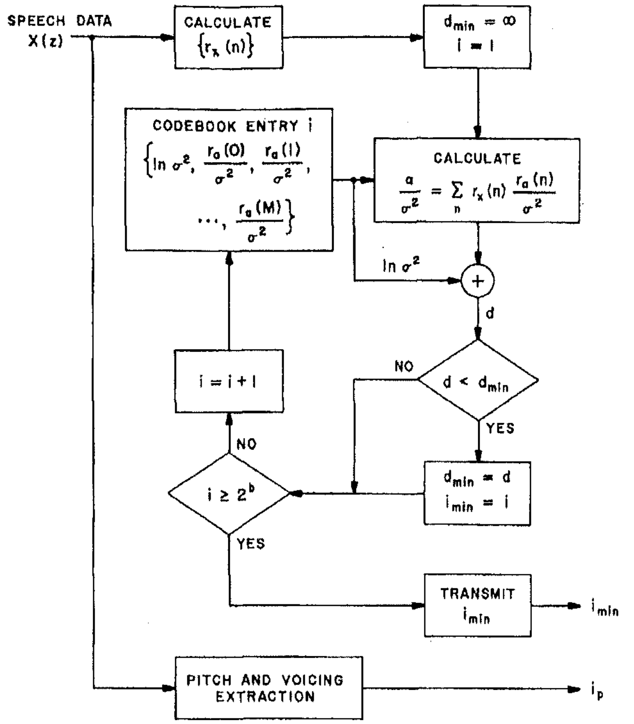

Fig. 5. Analysis structure for suboptimal vector quantized speech processor.

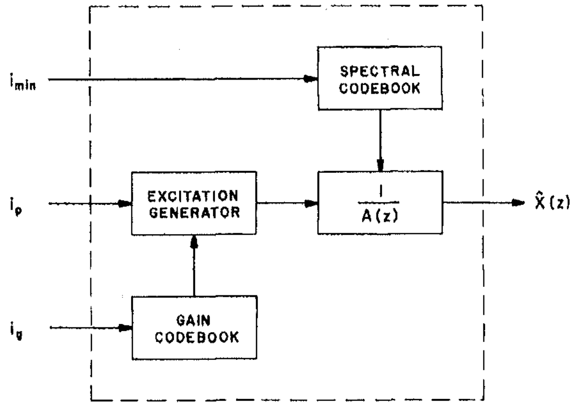

Fig. 6. Synthesis structure for suboptimal vector quantized speech processor.

V. An Experimental Comparison Between Vector and Scalar Quantization

It is important to note that nowhere in the theory are we assured of receiving substantial bit savings when processing actual speech data. The benefits, if any, must be demonstrated by way of actual experimentation. The results pre-

Exhibit X Page 453

sented in this section show that the benefits are substantial. These results are not intended to provide an exaustive presentation of the vector quantizer's capabilities and limitations. Rather, we describe one experiment with a substantial database which shows the dramatic benefits obtainable (in terms of bits/frame reduction for a given distortion level) to motivate additional research in this area.

A reference database of conversational speech segments from five male speakers was chosen with each speech segment consisting of approximately 20 s of speech. A 20 s test segment from a male speaker not included in the reference set was also chosen for test purposes.

The data were sampled at 6.5 kHz and digitally preemphasized. In addition, a sharp cutoff (200 Hz) high-pass filter was applied to eliminate very low-frequency energy. Previous tests have shown that if any A/D bias exists, for example, analysis coefficients will be used in representing the very low-frequency nonspeech behavior.

For this study we chose a preemphasis factor of 0.9, and filter order of 10. Analysis windows of 128 samples were chosen with a frame shift of 128 samples. A total of 5396 reference frames was obtained from analyzing 1 min, 48 s of speech. Although separate quantization tables could be obtained for voiced and unvoiced speech with possible further benefits, this experiment was based upon all frames for the reference data. Hamming windows were used.

The scalar quantizing method chosen was the uniform sensitivity quantization method proposed by Vishwanathan and Makhoul [4] and later studied by Gray and Markel [5]. The reflection coefficients are transformed into log area ratios [4] or inverse sine (theta) parameters [5] and then uniformly quantized on the basis of measured parameter sensitivities and parameter extreme values. The inverse sine transformation, which theoretically produces precisely constant sensitivity for the final coefficient, was used here. The procedure for actually measuring sensitivities and performing the bit allocation is presented elsewhere [5].

Because of computational and storage requirements for the full search optimal vector quantizer described earlier, it is of more practical interest to observe how well the suboptimal vector quantizer operates. Both the full search and binary tree searched suboptimal vector quantizer procedure were implemented to determine the tradeoff between search efficiency and distortion.

Computer programs were written in Fortran and implemented on a DEC VAX-11/780 computer system. The virtual memory capability of the VAX was found to be indispensable for this type of problem since main memory overflow occurs at about 8 bits/frame on a 16 bit computer because of the required data storage for efficient computation in the full search codebook generation.

Both the full search and binary search codebooks were obtained by the cell-splitting procedure starting with two cells, then four cells, etc., in a binary fashion to 10 bits or 1024 cells. We chose the 10 bit codebook limit for the spectral coefficients because of the database size of 5396 frames. The gain codebook was chosen to have 5 bits as this provided nearly as good quality as that obtainable at higher rates. With a standard 8 bits/frame for voicing and pitch, our highest rate

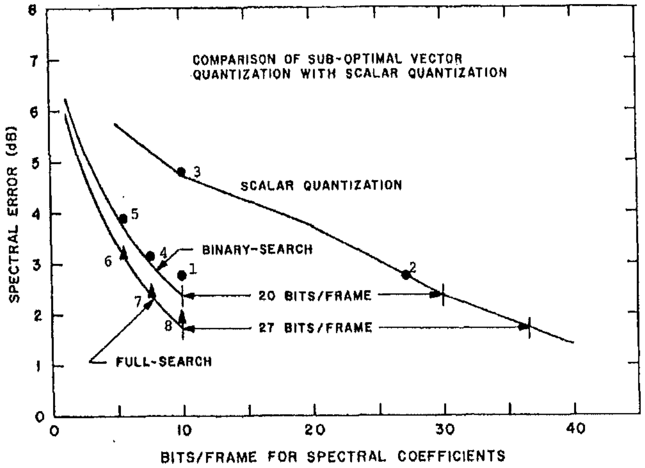
COMPARISON OF SUB-OPTIMAL VECTOR QUANTIZATION WITH SCALAR QUANTIZATION
SCALAR QUANTIZATION
BINARY-SEARCH
20 BITS/FRAME
27 BITS/FRAME
FULL-SEARCH
SPECTRAL ERROR (dB)
BITS/FRAME FOR SPECTRAL COEFFICIENTS
0
10
20
30
40
1
2
3
4
5
6
7
8

Fig. 7. Comparison of suboptimal vector quantization with scalar quantization.

system required $10 + 5 + 8 = 23$ bits per frame, or $20 \times 50 = 1150$ bits/s. The full search suboptimal vector quantizer codebook determination required 3 h, 20 min of stand-alone CPU time on the VAX system. The binary search procedure required 4 min, 47 s of stand-alone CPU time. At the completion of the binary tree searched codebook generation there were three empty cells for the 9 bit quantizer, 26 empty cells for the 10 bit codebook, and no empty cells for the full search quantizer.

For comparison with previously published results [7], the Itakura-Saito distortion measure, obtained from the codebook generation, was transformed into approximate rms log spectral error measurements using the result from (12) that for small distortion, the Itakura-Saito distortion measure is approximately one-half the mean-square log spectral deviation. The results of this experiment are shown in Fig. 7. The scalar quantization curve was obtained over the wide range of 6 bits/frame to 40 bits/frame for the spectral coefficients. The suboptimal vector quantizer was designed in increments of 1 bit/frame from 1 bit/frame to 10 bits/frame for the spectral coefficients. Both full search and binary search methods were implemented. For both the scalar and vector quantizer, the curves represent distortions measured during the reference codebook or reference quantization table design.

For a 10 bits/frame full search vector quantizer, the measured distortion is approximately 1.8 dB. The equivalent distortion point for the scalar quantizer is at approximately 37 bits/frame resulting in a difference of 27 bits/frame or a 73 percent reduction in bit rate for the equivalent distortion. When one places into perspective the fact that the field of linear prediction analysis has been concerned about bit rate reduction for a given distortion level for nearly 10 years [3], [6], [11], [12], and until recently [7] the differences between various coding schemes were 3–4 bits/frame for a given rms log spectral value, these results are remarkable.

Now, considering the suboptimal vector quantizer in more detail, we see that, whereas the full search and binary search procedures are identical at 1 bit/frame, the relative distortion of the binary search approach increases as bits/frame increase. At 10 bits/frame the distortion for the binary search is approxi-

Exhibit X Page 454

mately 0.6 dB higher than for the full search method. Stated differently, the binary search procedure requires roughly 2 bits/frame more than the full search procedure at 8 bits/frame. This would appear to be a very reasonable tradeoff for practical implementation since $2^{10} = 1024$ residual energy calculations per frame (each requiring $M + 1$ multiply/add's) are reduced to $2 \times 10 = 20$ residual calculations. With respect to the binary search procedure, at 10 bits/frame, equal distortion for scalar quantization requires 20 additional bits/frame.

Although objective comparisons such as the rms spectral error or distortion can be very useful for comparing different speech processing systems, it is well known that subjective evaluation is ultimately most important. We therefore decided to perform at least some informal perceptual testing to compare vector and scalar quantization.

A 20 s test segment described earlier was processed resulting in 600 frames of "open-test" data. In addition, 600 frames of the reference set were processed to define "closed-test" data.

The open-test samples (indicated by closed circles) were processed with the full search vector quantizer for 6, 8, and 10 bits/frame. Scalar quantization was implemented for 28 bits/frame and 10 bits/frame. The former choice allowed us to compare the perceptual differences between the two methods for a constant distortion value. The latter choice allowed us to judge the perceptual differences between scalar and vector quantization for an identical number of bits/frame.

In addition, we obtained distortion values for the closed-test samples using the full search procedure (indicated by solid triangles) to provide a comparison with open-test samples. Numbers to the side of each dot or triangle indicate the specific synthesized speech samples.

Samples 1 and 2 are judged perceptually to be very similar, both having a distinct "warble-like" characteristic with high intelligibility.

It is important to note that the distortion level for this open-test was 2.6 dB, resulting in a scalar representation of about 28 bits/frame. In the present government ANDVT system, for example [20], 42 bits/frame are used, resulting in a distortion or average rms log spectral difference of around 1.0 dB.

When sample 1 is compared with sample 3, the perceptual differences are obvious and dramatic. For the same number of bits/frame the scalar quantizer causes an increased distortion to about 4.8 dB and a very obvious decrease in intelligibility.

It is interesting to compare the closed-test samples (6, 7, 8) with the open-test samples (1, 4, 5). In general, as bits/frame decrease, increased "warble" occurs (along with increased distortion as an objective measure). As one might expect, the closed-set samples are preferred perceptually. However, the differences are not great. We have experimentally determined that it is quite difficult to perceptually distinguish between samples whose distortion is less than about 0.5-0.7 dB even with formal *A-B* comparisons over tightly coupled headphones.

VI. Summary/Future Research

A procedure for vector quantization of speech has been developed and demonstrated to have a dramatic advantage over scalar quantization when compared on an rms log spectral measure basis. Informal listening has verified that equivalent rms log spectral measures are also, roughly speaking, perceptually equivalent. That is, at a specified dB level such as 1.8 dB, no marked preference occurs between a 35 bits/frame scalar quantized speech sample and a 10 bit full search vector quantized sample.

TABLE I
Possible Tree Structures Ranging from Binary Search to Full Search and Corresponding Number of Residual Energy Evaluations

Number of Branching Levels l	Number of Branches at Each Node n	Number of Residual Energy Evaluations nl
12	2	24 (binary search)
6	4	24
4	8	32
3	16	48
2	64	128
1	4096	4096 (full search)

Many future research areas should be considered. We will attempt to summarize here what we believe to be some of the most interesting and fruitful of these research areas. First, methods for reducing the differences in distortion between the binary search and full search method should be investigated. As the number of bits increases, the degradation of binary tree search over the exhaustive search in terms of spectral deviation was seen to increase. This is to be expected as one moves further down the tree branches.

One approach to improvement is to use trees that are not so deep as a binary tree with more branches at each node. In particular, for a case of 12 bits, one can think of the exhaustive search as a tree with only one branching node, with 2^{12} branches coming out, whereas the binary tree has a set of 12 branching levels with only two branches coming out of each node. One could easily use 6 branching levels with 4 branches at each node (as $2^{12} = 4^6$), or 4 branching levels with 8 branches at each node (as $2^{12} = 8^4$), etc.

If we let l be the number of branching levels and n be the number of branches at each level, the total number of levels will be

$$n^l = 2^b$$

where b is the number of bits in the code word. The total number of residual energy evaluations at each branching level will be n, and as there are l levels, the total number of residual energy evaluations will be nl. Table I illustrates the possibilities for a 12 bit codebook.

In this case it appears that by using $l = 6$ and $n = 4$, performance could be improved without increasing the number of residual energy evaluations.

A great deal of study needs to be done concerning the codebook generation and the local optimality of the results. In particular, there are a number of methods for initializing the codebook generation and carrying out the cell splitting. Different techniques should be compared so that both rate of convergence and size of resultant distortion can be compared.

Another topic of great concern in codebook generation is methods of treating the *empty cell* problem, where one attempts to split a cell with only one speech frame in it. In the fully optimal case, this can be handled by bypassing the split and instead using the extra code word for handling the worst speech frame in the data space (as one possibility). This approach fails for the binary tree, and at present it appears that one must either waste some of the possible tree branches by "pruning," or restart the algorithm with a different set of initial conditions. While the empty cell problem is rare, particularly when a sufficiently large database is used for a training sequence (at least 50 times the total number of cells), it can occur.

Other possible studies are listed below.

1) The use of a binary codebook as an initialization for the optimal codebook.

2) The use of a "standard" codebook, based on a variety of speakers, as an initialization for a single speaker or speaker group codebook. While this approach is straightforward for the optimal codebook, there are a number of problems in applying it to the binary tree searched codebook.

3) The use of short training sequences for initialization of the codebook.

4) The use of interframe memory to further reduce the bit rate. Simulations have shown that most code words are almost always followed by a code word from a small subset of the entire codebook. Thus, for example, a lower bit rate might be achieved by searching only a "conditional codebook" on the basis of the last code word sent and then sending the index of the code word from the smaller codebook.

5) If a low rate "side channel" is available, then the systems described here could be made adaptive by occasionally updating and improving the codebook on the basis of the data being compressed. The new, improved code words would be sent to the receiver at a much slower rate than the speech data.

6) Performance of a subjective study of the codebooks produced by the algorithm to determine whether they relate to physically perceivable "cluster" points of speech sounds such as voiced, unvoiced, male, and female. Such a connection might aid in improving design techniques for the binary search.

The suboptimal approach described here leads to a theoretical problem in that speech frames are all normalized so that frames of silence are treated as valid speech frames unless removed by preprocessing. Further study of this preprocessing and the theoretical interpretation of resultant spectral deviations is needed. The approach will probably attempt to use an elementary thresholding silence detector, which will artificially zero out the frames corresponding to silence.

While it may not be practically feasible for regular use, it is interesting to consider actually implementing the fully optimal system (including the gain term) to find a baseline for the spectral distortion and establish how much is lost in the suboptimal case where the gain is separated out. Although intuitively it seems that the effects of separating or not separating the gain term should be small, this conjecture should be verified or disproved. This topic is being actively studied at present and preliminary results are reported in [18].

Finally, a basic problem in using vector quantization for medium bit rates lies in the storage required and the size of the training sequence. For example, if we needed a 20 bit codebook to achieve a sufficiently low distortion, this would require the storage of $2^{20} = 1\,048\,576$ code words. In addition, if we required 20 frames for each code word to achieve any reasonable clustering for the reference set, this would require well over 20 million speech frames in a training sequence. The calculations required to produce this codebook would tax the capabilities of any computer and strain any researcher's computer budget.

To achieve some sort of compromise between vector and scalar quantization, some partitioning of the information in the filter description is needed. A number of possibilities exist conceptually, although many may not be practical. For example, one approach in analyzing speech would be to find the full precision model filter, use a root solver to factor it according to some rule, and finally separately code the factors for the filter. Such a procedure might be possible using two codebooks, perhaps of 12 and 11 bits, where the total number of individual code words is decreased while the total number of bits would probably have to increase. The basic shortcoming with this approach lies in the necessity of using a root finder for factoring the model filters.

A second approach, also suboptimal, is to replace the linear predictor inverse filtering operation with a cascade of two inverse filters: the first to remove the resonant information and the second for fine resolution spectral shaping. This would allow the two filters to be individually coded, again using fewer stored code words but probably more bits.

A third approach would apply a linear predictor on the basis of a cascade of second-order sections, such as described by Jackson, Rao, and Wood [19]. These filters are obtained through an iterative type of solution and would readily lead to the factoring described in the preceding section.

A forth approach would be to partition the information into frequency bands, where the input signal would be passed through a set of filters. The outputs of these filters would cover a frequency range where there could be no more than two resonances. Each of these outputs would then be applied to a linear predictor analysis of relatively low order, each model filter using a much smaller number of bits than the overall total. Some research has been published in this area under the name of "piecewise linear predictive coding" [22].

Yet another approach might be to separately evaluate the size of a codebook needed for voiced speech only and unvoiced speech only. At present all speech is mixed together and normalized so that all frames have the same weight, regardless of gain level. A previous study has shown that unvoiced frames can be represented with substantially fewer coefficients and bits/frame than voice speech frames [23].

Appendix: Derivations of Equations

We present here the derivations in abbreviated form of the equations in the main text. Earlier, more formal derivations are given in [10]. As the material of Section II-A consists only of definitions and referenced results, we start with Section II-B.

Equation (13) is identical to [1, eq. (6.10)]. Using the time domain relation for the residual energy (8), it can also be expressed in the form

$$d[|X|^2; |G|^2] = \frac{1}{\sigma^2} \sum_n r_x(n)\, r_a(n) + \ln\,[\sigma^2/\alpha_\infty] - 1. \quad \text{(A1)}$$

Exhibit X Page 456

By direct substitution one can also write

$$d[|G_M|^2; |G|^2] = \frac{1}{\sigma^2} \sum_n r_M(n)\, r_a(n) + \ln [\sigma^2/\alpha_M] - 1 \tag{A2}$$

where we have used the property that

$$\int_{-\pi}^{\pi} \ln [|G_M|^2] \frac{d\theta}{2\pi} = \ln (\alpha_M) - \int_{-\pi}^{\pi} \ln [|A_M|^2] \frac{d\theta}{2\pi} = \ln (\alpha_M) \tag{A3}$$

for the integral in the middle expression is zero [1, Sect. 6.2.1].

Equations (A1) and (A2) can be subtracted. Using the correlation matching property (9) the summations vanish, and one finds

$$d[|X|^2; |G|^2] - d[|G_M|^2; |G|^2] = \ln (\alpha_M/\alpha_\infty).$$

But from (11), $\ln(\alpha_m/\alpha_\infty)$ is the minimum distortion, $d[|X|^2 |G_M|^2]$, thus giving (14).

Equation (15) follows from a direct substitution. Starting with (13), for $G(z) = \sigma/A(z)$,

$$d[|X|^2; \sigma^2/|A|^2] = \alpha/\sigma^2 + \ln (\sigma^2/\alpha_\infty) - 1, \tag{A4}$$

so that by direct substitution

$$d[|X|^2; \alpha/|A|^2] = \ln (\alpha/\alpha_\infty). \tag{A5}$$

Then from the definition of (10), by substitution of the constants α and σ^2,

$$d[\alpha; \sigma^2] = \alpha/\sigma^2 - \ln (\alpha/\sigma^2) - 1. \tag{A6}$$

By inspection, adding the right-hand sides of (A5) and (A6), we obtain the right-hand side of (A4), hence (15) follows.

Equations (16) and (17) are definitions. Equation (18) follows from (8) and the even property of autocorrelation sequences, for

$$\alpha = \sum_{n=-\infty}^{\infty} r_x(n)\, r_a(n) = r_x(0)\, r_a(0) + 2 \sum_{n=1}^{\infty} r_x(n)\, r_a(n). \tag{A7}$$

As the filter coefficients are limited to $M+1$ in number $\{a_0, a_1, \cdots, a_M\}$, the autocorrelation sequence $\{r_a(n)\}$ terminates at $n = M$:

$$r_a(n) = \begin{cases} \sum_{k=0}^{M-n} a_k a_{k+n} & \text{for } n = 0, 1, \cdots, M \\ 0 & \text{for } n > M, \end{cases} \tag{A8}$$

and thus (A8) is equivalent to (18).

Equation (19) follows from (A8), and (20) is the common formula for the short-term autocorrelation function for a truncated sequence.

For the centroid calculation of (21), we start with the definition of (16), rewriting the individual terms, $d[|X_k|^2, |G|^2]$ using superscripts of "k" for the "kth" frame residual energy and one-step prediction error. From (13), (4), and (6),

$$\begin{aligned} d[|X_k|^2; |G|^2] &= \alpha^k/\sigma^2 + \ln (\sigma^2) - \ln (\alpha_\infty^k) - 1 \\ &= \frac{1}{\sigma^2} \int_{-\pi}^{\pi} |X_k|^2 |A|^2 \frac{d\theta}{2\pi} + \ln (\sigma^2) \\ &\quad - \int_{-\pi}^{\pi} \ln |X_k|^2 \frac{d\theta}{2\pi} - 1. \end{aligned} \tag{A9}$$

If this expression is now summed from $k = 1$ to $k = L$, and the definition of the average $|\overline{X}|^2$ of (17) is used, then

$$\begin{aligned} D &= \frac{L}{\sigma^2} \int_{-\pi}^{\pi} |\overline{X}|^2 |A|^2 \frac{d\theta}{2\pi} + L \ln (\sigma^2) \\ &\quad - \int_{-\pi}^{\pi} \sum_{k=1}^{L} \ln |X_k|^2 \frac{d\theta}{2\pi} - L. \end{aligned} \tag{A10}$$

As in (A9), we also have the definition

$$\begin{aligned} d[|\overline{X}|^2; |G|^2] &= \frac{1}{\sigma^2} \int_{-\pi}^{\pi} |\overline{X}|^2 |A|^2 \frac{d\theta}{2\pi} + \ln (\sigma^2) \\ &\quad - \int_{-\pi}^{\pi} \ln |\overline{X}|^2 \frac{d\theta}{2\pi} - 1, \end{aligned}$$

which can be combined with (A10) directly to yield the result of (21), where the constant u is given by

$$u = L \int_{-\pi}^{\pi} \left[\ln |\overline{X}|^2 - \frac{1}{L} \sum_{k=1}^{L} \ln |X_k|^2 \right] \frac{d\theta}{2\pi}. \tag{A11}$$

The constant u is thus a measure of the "spread" within the cluster. The integrand in (A11) can also be looked at as the log of the ratio of the arithmetic mean of the separate spectra to their geometric mean. The logarithm is then averaged over θ.

Acknowledgment

The authors would like to express appreciation to Dr. D. Y. Wong of STI for his careful reading of the manuscript, F. Juang of STI for performing several of the experiments, G. Rebolledo of Stanford University, Stanford, CA, for helpful discussions, and Dr. D. Lee of Stanford for his assistance.

References

[1] J. D. Markel and A. G. Gray, Jr., *Linear Prediction of Speech.* New York: Springer-Verlag, 1976.

[2] J. Makhoul, "Linear prediction–A tutorial review," *Proc. IEEE*, vol. 63, pp. 561–580, Apr. 1975.

[3] F. Itakura and S. Saito, "On the optimum quantization of feature parameters in the PARCOR speech synthesizer," in *Proc. IEEE Conf. Speech Commun. Processing*, New York, 1972, pp. 434–437.

[4] R. Viswanathan and J. Makhoul, "Quantization properties of transmission parameters in linear predictive systems," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 309–321, June 1975.

[5] A. H. Gray, Jr. and J. D. Markel, "Quantization and bit allocation in speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 459–473, Dec. 1976.

[6] A. H. Gray, Jr., R. M. Gray, and J. D. Markel, "Comparison of optimal quantization of speech reflection coefficients," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 9–23, Feb. 1977.

[7] J. D. Markel and A. H. Gray, Jr., "Implementation and comparison of two transformed reflection coefficient scalar quantization methods," this issue, pp. 575–583.

[8] M. R. Sambur, "An efficient linear prediction vocoder," *Bell Syst. Tech. J.*, Dec. 1975.

[9] E. Blackman, R. Viswanathan, and J. Makhoul, "Variable-to-fixed rate conversion of narrow-band LPC speech," in *Conf. Rec., 1977 IEEE ICASSP Conf.*, ASSP 77CH1197-3, 1977, pp. 409–412.

[10] R. M. Gray, A. Buzo, A. H. Gray, Jr., and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 367–376, Aug. 1980.

[11] F. Itakura and S. Saito, "Analysis synthesis telephone based upon the maximum likelihood method," in *Conf. Rec., 6th Int. Congr. Acoust.*, Y. Yonasi, Ed., Tokyo, Japan, 1968.

[12] Y. Matsuyama, A. Buzo, and R. M. Gray, "Spectral distortion measures for speech compression," Stanford University, Stanford, CA, ISL Rep. 6504-3, Apr. 1978.

[13] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.

[14] L. A. Buzo de la Pena, "Optimal vector quantization for linear predictive coded speech," Ph.D. dissertation, Stanford University, Stanford, CA, Aug. 1978.

[15] S. P. Lloyd, "Least squares quantization in PCM," Bell Lab., Murray Hill, NJ, Tech. Rep., 1957.

[16] A. H. Gray, Jr. and J. D. Markel, "Distance measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 380–391, Oct. 1976.

[17] J. D. Markel and S. B. Davis, "Text-independent speaker recognition from a large linguistically unconstrained time-spaced data base," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 74–82, Feb. 1979.

[18] R. M. Gray, A. H. Gray, Jr., G. Rebolledo, and J. E. Shore, "Rate distortion speech coding with a minimum discrimination information distortion measure," submitted to *IEEE Trans. Inform. Theory*.

[19] L. Jackson, R. M. Rao, and S. L. Wood, "Parameter estimation by linear prediction in cascade form," in *Conf. Rec., 1977 IEEE Conf. Acoust., Speech, Signal Processing*, ASSP 77CH1197-3, 1977, pp. 727–731.

[20] G. S. Kang, L. J. Fransen, and E. L. Kline, "Multirate processor (MRP) for digital voice communications," Naval Res. Lab., Washington, DC, NRL Rep. 8295, pp. 76–80, Mar. 1979.

[21] R. M. Gray, J. C. Kieffer, and Y. Linde, "Locally optimal block quantizer design," *Inform. Contr.*, to be published.

[22] J. E. Roberts and R. H. Wiggins, "Piecewise linear predictive coding (PLPC)," in *Proc. 1976 IEEE Int. Conf. Acoust., Speech, Signal Processing*, IEEE Cat. No. 76CH1067-8 ASSP, pp. 470–473, Apr. 1976.

[23] D. Y. Wong and J. D. Markel, "An intelligibility evaluation of several linear prediction vocoder modifications," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 424–435, Oct. 1978.

Andrés Buzo (S'76–M'78), for a photograph and biography, see p. 376 of the August 1980 issue of this Transactions.

Augustine H. Gray, Jr. (S'56–M'65–SM'79), for a photograph and biography, see p. 376 of the August 1980 issue of this Transactions.

Robert M. Gray (S'68–M'69–SM'77–F'80), for a photograph and biography, see p. 376 of the August 1980 issue of this Transactions.

John D. Markel (M'72–SM'77) was born in Wichita, KS, on January 20, 1943. He received the B.S.E.E. degree from Kansas State University, Manhattan, in 1965, the M.S.E.E. degree from Arizona State University, Tempe, in 1968, and the Ph.D. degree from the University of California, Santa Barbara, in 1970.

From 1965 to 1969, he was employed at Motorola Government Electronics Division, Scottsdale, AZ. In 1969 he joined the Speech Communications Research Laboratory (SCRL), Santa Barbara, CA. He was actively involved in various speech research topics and became Vice President in 1972. During this period, he was also a Consultant for government and commercial organizations and developed speech and signal processing short courses. In 1977 he founded Signal Technology, Inc., Santa Barbara, CA, where he is President. He is presently involved in speech processing research and consulting, and management of various business areas of research contract work, signal processing, software development, consulting, and computer services.

Dr. Markel and Dr. A. H. Gray, Jr., are coauthors of the book *Linear Prediction of Speech* (New York: Springer-Verlag, 1976) and have received several awards, including the 1977 IEEE ASSP Achievement Award for their contributions to the development of linear prediction techniques.

Exhibit X Page 458